CS583A: Course Project

Yao Xiao

May 19, 2019

1 Summary

We pariticipet an inactive with late submission competition of Humpback Whale Identification.

The final model We choose is Siamese Neural Network. A Siamese Neural Network is composed of two parts. A Convolutation Neural Network (CNN) transforms an input image into a vector of features describing the whale called branch model. Another Convolutation Neural Network (CNN) used to compare the feature vectors from the CNN and decide if the whales match or not called head model. During training, We will always have pairs of images as input. We will feed those pairs into one same ConvNets layer, which outputs a fixed length feature vector (e.g. a 512 vector) to represent features of that image. Then we will calculate the distance between the two output feature vectors from each image, and output a probability (e.g. through a sigmoid layer) to indicate the chance of two images being in the same category. The SNN model takes 384×384 images as input and outputs the class labels.

We also use the bounding box model to crop image of whale fluke. Because it could give better results than simply feeding the original image during training. We removed majority of the background noise, such as water, flying bird, etc, and let the model concentrate on the more important details, which could improve the model's performances and given the amount of training data is relatively small. We build a CNN model (similar to VGG16 architecture) and let it output four floating numbers, corresponding to the x and y of two corners of the rectangular box.

We implement the convolutional neural network using Keras and run the code on a MacBook Pro with one Intel i7 CPU and 32 GB memory. Performance is evaluated on the MAP(Mean Average Precision).

In the public leaderboard, the score is 0.85538; since the competition has already finished, this score will rank as same as 495 among the 2131 teams. In the private leaderboard, the score is 0.86829; this score will rank as same as 598 among the 2131 teams.

2 Problem Description

Problem. The problem is to identify different species of whales based on whale tails' photos. This is a image recognition problem. The competition is at `https://www.kaggle.com/c/humpback-whale-identification/overview`.

Data. The data are different size of JPEG images. The number of training samples is $n = 25,361$. The number of test samples is $n = 7,960$. The number of classes is $5,005$.

Challenges. There are 4 challenges:

1. Each of the image has relatively high resolution in its original format. However, some outlier images, where either the photo dimension is very distorted, or the images are not displayed properly.

2. %40 classes have only one image.

3. 9,664 of all the whales belong to a class called new whale, which you can think of as unidentified whale. The interesting thing about new whale is that you cannot simply group those 9,664 images as one class for training some of them may belong to one category, but it's just we don't know.

4. All images are whale's tails so that different classes have little different in images. Features that models need to learn to tackle this task well are those tiny little differences on whales flukes.

3 Solution

Model. The model We finally choose is the Siamese Neural Network [1], dealing with one-shot problem. A description of SNN is online: `https://en.wikipedia.org/wiki/Siamese_network`.

Implementation.

1. Remove duplicate images

   Firstly, We calculate the Perceptual Hash of each images to identify the duplicate images. Then by using p2h and h2p functions to make sure each ID only have one images. The p2h dictionary associate a unique image id (phash) for each picture. The h2p dictionary associate each unique image id to the prefered image to be used for this hash.The prefered image is the one with the highest resolution, or any one if they have the same resolution.

2. Image preprocessing

   In this part We rotate the image if it is in the rotate set. Transform images to black and white and apply an affine transformation.

3. Build bounding box model

   While many of the whale pictures in the dataset are already cropped tight around the whale fluke, in some images the whale fluke occupies only a small area of the picture. Zooming in the relevant part of the picture provides greater accuracy to a classification model. Build a convolutional neural network (CNN) model to implement bounding box. Training dataset is 1000 images and testset is 200 images.

4. Pretrained model

   We used the pretrained weights from martinpiotte[2]: standard model and bootstrap model. The standard model is trained on the smallest training set, and thus has more potential for overfitting. The bootstrap model is trained on more data, however the tagging accuracy is lower since the bootstrap data is only 93% accurate. The best performance of two models weight is a weight of 0.45 for the standard model and 0.55 for the bootstrap model.

5. Build Siamese Neural Network model

   The SNN compares two images and decides if the two images are taken from the same whale, or different whales. By testing each image from the test set against every image from the training set, the most likely whales can be identified by sorting the pictures in likelihood for a match. A SNN model is composed of two parts.

   Branch model transforms an input image into a vector of features describing the whale. The branch model is a regular CNN model composed of 6 blocks, each block processing maps with smaller and smaller resolution, with intermediate pooling layers.

   - Block 1 - 384x384
   - Block 2 - 96x96
   - Block 3 - 48x48
   - Block 4 - 24x24
   - Block 5 - 12x12
   - Block 6 - 6x6

   Head model compares the feature vector from the branch model to decide if the pictures show the same or different whales.

6. Training

   We want the Siamese Neural Network to pick the one correct whale from all the possible whales from the training set. While scoring the correct whale high, it must simultaneously score all other whales lower. It is not enough to have a random whale score low. To force all the other whales to a low probability, the training algorithm presents pairs of pictures with increasing difficulty, as evaluated by the model at any given time. Essentially, we focus the training on pairs that he model is getting wrong, as a form of adversarial training.

   Half the examples used during training are for pair of images. For each whale of the training set, compute a derangement of its pictures. Use the original order as picture A, and the derangement as picture B. This creates a random number of matching image pairs, with each image taken exactly two times.

   Training lasts 400 epochs, with the following quantities changing as the training progresses:

   - Learning rate
   - L2 regularization
   - Constant K measuring the scale of the random component of the score matrix used to match similar images to construct tough training cases.

7. Test

   The basic strategy is this. For each image from the test dataset:

- If the image duplicates one or more training set images, add the whales (possibly more than one) from the training image as the top candidates.
- For each image not new whale from the training set, compute the image score, which is the model score for the image pair.
- For each whale from the training set, compute the whale score as the maximum image score for this whale.
- Add new whale with a fixed new whale score of 'threshold'.
- Sort the whales in decreasing score.

We implement the SNN model and bounding box model using Keras with TensorFlow as the backend. Our code is available at `https://github.com/yaoxiao21/Humpback-Whale-Identification/`. We run the code on a MacBook Pro with one Intel i7 CPU and 32 GB memory. It takes 3 hours to train the SNN model and 22 hours to train the bounding box model.

Settings. Training the large model from random weights is difficult. In fact, if the model is initially fed examples that are too hard, it does not converge at all. In the context here, hard examples are similar images belonging to different whales. Pushed to the extreme, it is possible to construct a training dataset for which pairs of pictures of different whales appear (to the model) more similar than pairs of pictures from the same whale, making the model learn to classify similar images as different whales, and dissimilar images as same whales.

To prevent this, early training is executed with a large value of K, making the negative examples essentially random pairs of pictures of different whales. As the model ability to distinguish between whales increases, K is gradually reduced, presenting harder training cases. Similarly, training starts with no L2 regularization. After 250 epochs, trainings accuracy is fantastic, but it also grossly overfits. At this point, L2 regularization is applied, learning rate is reset to a large value and training continues for an additional 150 epochs.

Advanced tricks. The assembly strategy consist in compute a score matrix (or dimension test by train) that is a linear combination of the standard and bootstrap model. Generation of the submission using the score matrix is unchanged. Trial and error suggest a weight of 0.45 for the standard model and 0.55 for the bootstrap model.The resulting ensemble as an accuracy of 0.78563 using a threshold of 0.92.

Without using this assembly strategy we only got the 0.697 of score.

Cross-validation. The following Figure 1 shows the exact schedule used for the learning rate (LR), L2 regularization (L2) and randomized score matrix (K). Also note that the score matrix for the Linear Assignment Problem is computed at every 5 epochs starting with epoch 10.

Epochs	LR	K	L2
1-10	64.10^{-5}	$+\infty$	0
11-15	64.10^{-5}	100.00	0
16-20	64.10^{-5}	63.10	0
21-25	64.10^{-5}	39.81	0
26-30	64.10^{-5}	25.12	0
31-35	64.10^{-5}	15.85	0
36-40	64.10^{-5}	10.00	0
41-45	64.10^{-5}	6.31	0
46-50	64.10^{-5}	3.98	0
51-55	64.10^{-5}	2.51	0
56-60	64.10^{-5}	1.58	0
61-150	64.10^{-5}	1.00	0
151-200	16.10^{-5}	0.50	0
200-240	4.10^{-5}	0.25	0
241-250	10^{-5}	0.25	0
251-300	64.10^{-5}	1.00	2.10^{-4}
301-350	16.10^{-5}	0.50	2.10^{-4}
351-390	4.10^{-5}	0.25	2.10^{-4}
391-400	10^{-5}	0.25	2.10^{-4}

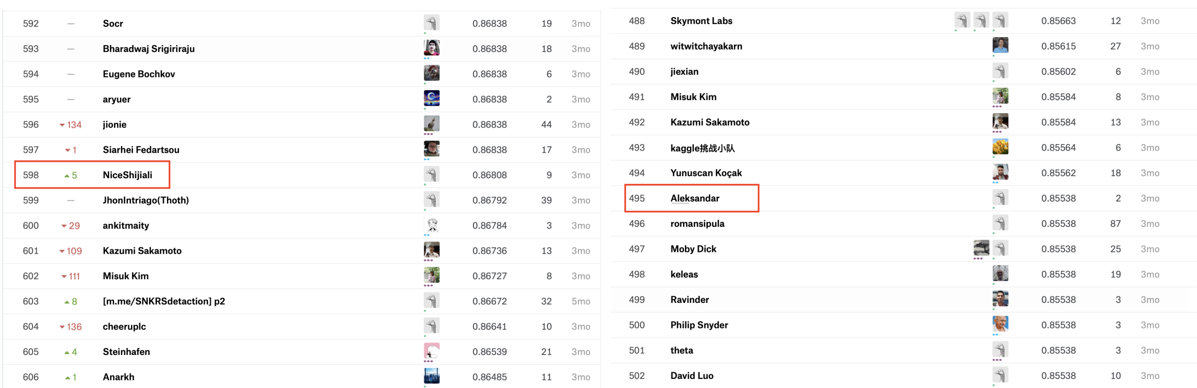
Figure 1: Table

4 Compared Methods

We compare our model with baseline model CNN. We use 25361 images as training dataset. The CNN model takes 100×100 images as input and outputs the class labels. We choose categorical crossentropy as loss function and run in 50 epochs. The result of MAP score is only 0.33530.

5 Outcome

We participated in an inactive competition. Our score is 0.85538 in the public leaderboard and 0.86829 in the private leaderboard. We rank 495/2131 in the public leaderboard and 598/2131 in the private leaderboard. The screenshots are in Figure 2.



592	—	Socr	0.86838	19	3mo
593	—	Bharadwaj Srigiriraju	0.86838	18	3mo
594	—	Eugene Bochkov	0.86838	6	3mo
595	—	aryuer	0.86838	2	3mo
596	▼ 134	jionie	0.86838	44	3mo
597	▼ 1	Siarhei Fedartsou	0.86838	17	3mo
598	▲ 5	NiceShijiali	0.86808	9	3mo
599	—	JhonIntriago(Thoth)	0.86792	39	3mo
600	▼ 29	ankitmaity	0.86784	3	3mo
601	▼ 109	Kazumi Sakamoto	0.86736	13	3mo
602	▼ 111	Misuk Kim	0.86727	8	3mo
603	▲ 8	[m.me/SNKRSdetaction] p2	0.86672	32	5mo
604	▼ 136	cheeruplc	0.86641	10	3mo
605	▲ 4	Steinhafen	0.86539	21	3mo
606	▲ 1	Anarkh	0.86485	11	3mo

(a) Private leaderboard.

488	Skymont Labs	0.85663	12	3mo
489	witwitchayakarn	0.85615	27	3mo
490	jiexian	0.85602	6	3mo
491	Misuk Kim	0.85584	8	3mo
492	Kazumi Sakamoto	0.85584	13	3mo
493	kaggle挑战小队	0.85564	6	3mo
494	Yunuscan Koçak	0.85562	18	3mo
495	Aleksandar	0.85538	2	3mo
496	romansipula	0.85538	87	3mo
497	Moby Dick	0.85538	25	3mo
498	keleas	0.85538	19	3mo
499	Ravinder	0.85538	3	3mo
500	Philip Snyder	0.85538	3	3mo
501	theta	0.85538	3	3mo
502	David Luo	0.85538	10	3mo

(b) Public leaderboard.

Figure 2: Our rankings in the leaderboard.

References

[1] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2, 2015.

[2] martinpiotte. Whale recognition model, aug 2018.